POLITECNICO
MILANO 1863

Computer Science and Engineering

Code Inspection

February 2, 2017

Prof. Luca Mottola

Authors:

- ZHOU YINAN(Mat. 872686)
- ZHAO KAIXIN(Mat. 875464)
- ZHAN YUAN(Mat. 806508)

Contents

1 Classes Assigned

We have been assigned two classes :

- ProductionRun.java
- ViewerServletRequest.java

The namespace patten is :

```
../apache-ofbiz-16.11.01/applications/manufacturing/src/main/java/org/apache/ofbiz
/manufacturing/jobshopmgt/ProductionRun.java
```

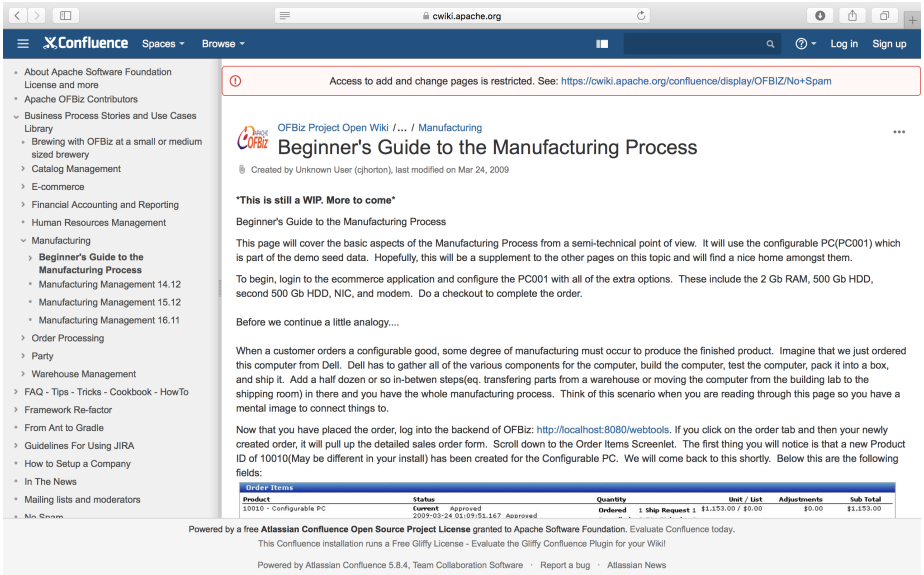
```
../apache-ofbiz-16.11.01/specialpurpose/birt/src/main/java/org/apache/ofbiz/birt
/report/servlet/ViewerServletRequest.java
```

2 Functional Role

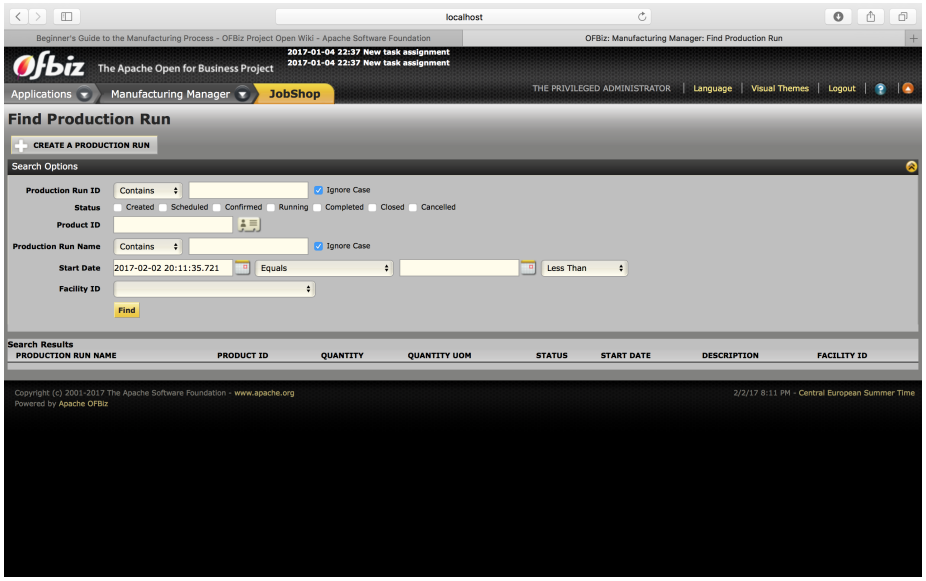
2.1 ProductionRun.java

Instead of directly looking into the code, we first examine the online ofbiz document to get information of this class. This class belongs to **Manufacturing** section. The link to the Online Documet.

The situation is described here. After a client makes order, configurable goods which our company provide require some type of manufacturing or production. If we do not have the requiring parts in our inventory, a production run is generated.



If we log into the ofbiz web application, we can examine the production run section. The ProductionRun class manages all the information of a certain production run activity.

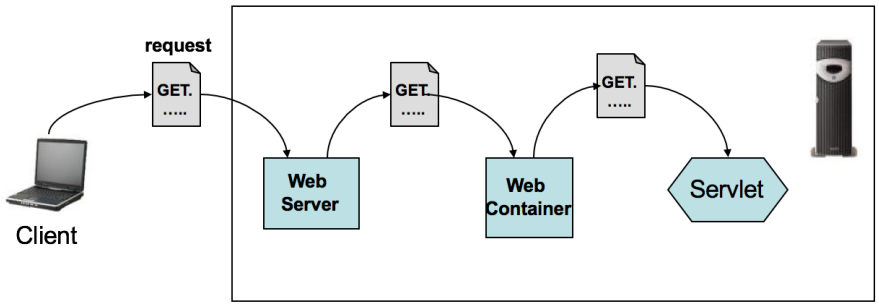


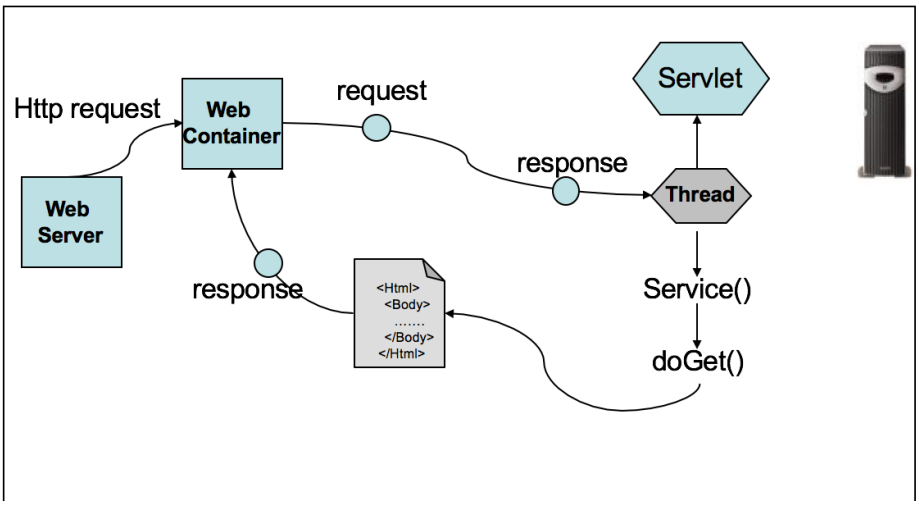
2.2 ViewerServeltRequest

By looking at the code, we find that ViewerservletRequest extends HttpServletRequestWrapper. A "HttpServletRequestWrapper" provides a convenient implementation of the HttpServletRequest interface that can be subclassed by developers wishing to adapt the request to a Servlet. Thus the role of this class is to represent a specific function of HttpServletRequest. More specifically, this function is getParameter();

Before looking into this function, let's recall what is a servlet. A servlet lives in a web container, and it is responsible for generating dynamic web contents. Servlet can be viewed as a special java class without main methods. After a client sends a HTTP request to the web server, the web container is responsible for :

- create an instance of a servlet
- call specific method of a servlet
- destroy a servlet





The web container knows which servlet to call because a servlet can have three names :

- Client knows URL name
- Deployer knows servlet secret internal name
- Actual java class name

The XML document is responsible for deployment.

```
<web-app>
 ……
 <servlet>
   <servlet-name>LoginServ</servlet-name>
    <servlet-class>com.Login</servlet-class>
  </servlet>

 <servlet-mapping>
   <servlet-name>LoginServ</servlet-name>
   <url-pattern>/Logon</url-pattern>
 </servlet-mapping>
 ……..
 ……..
</web-app>
```

Web.xml

Now let's look at what function role is this class. **ServletRequest** defines an object to provide client request information to a servlet. The servlet container creates a ServletRequest object and passes it as an argument to the servlet's service method. A **ServletRequest** object provides data including parameter name and values, attributes, and an input stream.
This java class file is used to form the parameter.

3 Check List

3.1 ProductionRun.java

3.1.1 Naming Conventions

3.1.2 Indention

3.1.3 Braces

3.1.4 File Organization

3.1.5 Wrapping Lines

3.1.6 Comments

3.1.7 Java Source Files

3.1.8 Package and Import Statements

3.1.9 Class ans Interface Declarations

3.1.10 Initialization and Declarations

3.1.11 Method Calls

3.1.12 Arrays

3.1.13 Object Comparison

3.1.14 Output Format

3.1.15 Computation, Comparisons, and Assignments

3.1.16 Exceptions

3.1.17 Flow of Control

3.1.18 Files

3.2 ViewerServletRequest

3.2.1 Naming Conventions

3.2.2 Indention

3.2.3 Braces

3.2.4 File Organization

3.2.5 Wrapping Lines

3.2.6 Comments

3.2.7 Java Source Files

3.2.8 Package and Import Statements

3.2.9 Class ans Interface Declarations

3.2.10 Initialization and Declarations

3.2.11 Method Calls

3.2.12 Arrays

3.2.13 Object Comparison

3.2.14 Output Format

3.2.15 Computation, Comparisons, and Assignments

3.2.16 Exceptions

1. The relevant exception is caught.

```
try {
reportFileUrl = FlexibleLocation.resolveLocation(reportParam, loader);
} catch (MalformedURLException e) {
Debug.logError(e, module);
}
if (reportFileUrl == null) {
throw new IllegalArgumentException("Could not resolve location to URL: "
+ reportParam);
}
```

The function in this code block tries to locate the file url. In case of wrong url and no file found, an exception is raised.

3.2.17 Flow of Control

No **switch** and **loop** in this file.

3.2.18 Files

This java class does not deal with file operations.